

## Extracting numbers from text strings, removing unwanted characters

Not being familiar enough with regular expressions, can somebody do the following?

Extract a number from a text string.

To keep it simple I want to remove all non-numeric characters and return the result as a number.

Typically keep all [0-9] and "." and "-" characters.

Presumably only keep 1 instance of decimal place and minus/hyphen.

I am importing numeric values from a file and typically the input string will only contain leading dollar sign, commas, digits, decimals and maybe a hyphen.

Note: I have looked at some regular expressions for numbers but of course these aren't removing unwanted characters.

### Answer

*Extract a number from a text string.*

*To keep it simple I want to remove all non-numeric characters and return the result as a number.*

```
% set input "As seen on TV, the cost is $19.95"
```

```
As seen on TV, the cost is $19.95
```

```
% regexp -- {-?\d+(\.\d+)?} $input number
```

```
1
```

```
% set number
```

```
19.95
```

```
% regexp -inline -- {-?\d+(?:\.\d+)?} $input
```

```
19.95
```

The pattern above: `{-?\d+(\.\d+)?}` translates to:

`-?` An optional hyphen;

`\d+` followed by one or more digits;

`(\.\d+)?` optionally followed by a single period and then one or more digits.

The quantifier "?" matches 0 or 1 occurrences, "+" matches 1 or more, and (not used in the above example), "\*" matches 0 or more. If your input could contain something like `-.235` then you would want a pattern of: `{-?\d*(\.\d+)?}` instead.

The period matches any (single) character, which is why we preceded it with a backslash to "escape it". If we didn't, then the pattern could match a non-number like (for example) `"-321-0"` or `"9z4"`. Braces are used instead of quotes to avoid having to "double up" the backslashes.

Since the first character of the regular expression was a hyphen, we preceed the pattern with `--` so that the regexp engine doesn't mistakenly try and interpret our pattern as (an invalid) switch.

In the regexp "non-capturing" parenthesis are used so that just the whole match is returned `{19.95}`. If we'd used regular paranthesis, as we did in the first example, `[regexp -inline]` would have returned a list made up of the whole match, and then what the paranthesis "captured" `{19.95 .95}`.

All of the ins and outs are documented on the `re_syntax` man page. See: [http://wiki.tcl.tk/re\\_syntax](http://wiki.tcl.tk/re_syntax)

*Typically keep all `[0-9]` and `"."` and `"-"` characters.*

*Presumably only keep 1 instance of decimal place and minus/hyphen.*

*I am importing numeric values from a file and typically the input string will only contain leading dollar sign, commas, digits, decimals and maybe a hyphen.*

To remove anything that isn't a digit, hyphen, or period, you could use `[regsub]` like this:

```
% set input "$1,000,000 is a lot of money"
$1,000,000 is a lot of money
% regsub -all -- {[^0-9.-]} $input "" number
21
% set number
1000000
```

If your data is consistant, and you need only remove a leading \$ and commas, you could also use `[string map]`:

```
% set input $1,995.99
$1,995.99
% set number [string map [list $ "" , ""] $input]
1995.99
```

If you only needed to remove a leading \$, you could use:

```
% set input $995.00
$995.00
% string trimleft $input $
995.00
% string range $input 1 end
995.00
```

## Answer 2

It may be helpful if you provide an example of typical input.

If the input string is delimited in some way (commas, colons, etc) then it's wisest to split on that delimiter.

You seem to be asking for a brute force regexp solution which may not be in your best interest. For example

```
set input {$1.23, $-5.00, $15.99}
```

```
regsub -all {[^0-9.-]} $input "" numbers
```

```
set numbers ;# ==> returns garbage "1.23-5.0015.99"
```

Please give us some more info about the typical input.

## Question

What would be the approach if in a single sting we will have multiple numeric values. How to get all numeric values?

For multiple strings having multiple numeric values, how to obtain them?

## Answer

You can use the -all option to [regexp]. With -inline you get a list of matching substrings.